

## AMENDMENTS TO THE CLAIMS

Please amend Claims 3, 12, 13, 16, and 17 as follows. Please cancel Claim 23. Please add Claim 24.

1. (canceled)

2. (canceled)

3. (currently amended) A multiple error correcting code (ECC) mechanism for use with transactions in a computer system, comprising:  
an ECC encoder, the ECC encoder applying a first ECC code to a first portion of a transaction and a second ECC code to a second portion of the transaction; and

a decoder that decodes the first and the second ECC codes, wherein the first portion of the transaction is a header packet, [[and]] wherein the second portion of the transaction comprises one or more data packets and wherein all of the data packets do not require an associated ECC code.

4. (original) The mechanism of claim 3, wherein the second ECC code is contained in a last data packet of the one or more data packets.

5. (previously presented) The mechanism of claim 3, wherein the second ECC code comprises a plurality of parity bits and one or more remaining ECC bits, wherein each data packet of the one or more data packets includes one parity bit of the plurality of parity bits, wherein a last data

packet of the one or more data packets includes the one or more remaining data bits, and wherein data may be processed without waiting for all of the one or more packets to be received if no parity bit errors exist.

6. (canceled)

7. (original) The mechanism of claim 3, wherein the second ECC code is a SEC-DED code.

8. (original) The mechanism of claim 3, wherein the second ECC code detects a wire failure.

9. (original) The mechanism of claim 3, wherein the transaction further comprises a third portion, and the mechanism further comprises a third ECC code.

10. (canceled)

11. (canceled)

12. (Currently amended) A method for protecting a transaction in a computer system by using a multiple error correcting code scheme, comprising:

applying a first ECC code to a first portion of the transaction;

applying a second ECC code to a second portion of the transaction;  
transmitting the transaction; and

decoding the first and the second ECC codes, wherein the first  
portion of the transaction is a header packet, ~~[[and]]~~ wherein the second  
portion of the transaction comprises one or more data packets and  
wherein all of the data packets do not require an associated ECC code.

13. (Currently Amended) The method of claim ~~10~~ 12, wherein the  
second portion of the transaction comprises one or more data packets.

14. (original) The method of claim 13, wherein the second ECC code is  
contained in a last data packet of the one or more data packets.

15. (original) The method of claim 13, wherein the second ECC code  
comprises a plurality of parity bits and one or more remaining ECC bits,  
wherein each data packet of the one or more data packets includes one  
parity bit of the plurality of parity bits, and wherein a last data packet of  
the one or more data packets includes the one or more remaining data  
bits.

16. (Currently Amended) The method of claim ~~10~~ 12, wherein the  
second ECC code is one of a SEC-DED code and a code that detects a wire  
failure.

17. (Currently Amended) The method of claim ~~10~~ 12, wherein the transaction further comprises a third portion, and the mechanism further comprises a third ECC code.

18. (original) The method of claim 12, further comprising:  
defining all transactions and an amount of information to be protected;  
defining interconnect widths and packet sizes; and  
determining a desired ECC capability.

19. (original) The method of claim 18, further comprising:  
listing all frequently-used, multiple packet transactions;  
determining a smallest transaction from the list; and  
determining if an alternative ECC may be used with the smallest transaction.

20. (original) The method of claim 19, further comprising:  
if the alternative ECC may be used, encoding all listed transactions using the alternative ECC; and  
if the alternative ECC may not be used:  
removing from the list, all transactions whose size equals that of the smallest transaction, and repeating the process until the list is empty.

21. (previously presented) The mechanism of claim 3, wherein the first ECC code is a single error correction-double error detection (SEC-DED) code.

22. (previously presented) The method of claim 12, wherein the first ECC code is a single error correction-double error detection (SEC-DED) code.

23. (Cancelled)

24. (New) A multiple error correcting code (ECC) mechanism for use with transactions in a computer system, comprising:

an ECC encoder, the ECC encoder applying a first ECC code to a first portion of a transaction and a second ECC code to a second portion of the transaction; and

a decoder that decodes the first and the second ECC codes, wherein the first portion of the transaction precede the second portion of the second transaction and wherein the second portion of the transaction comprises a plurality of data packet segments and wherein all of the data packet segments do not require an associated ECC code.